Table of contents

Learn the advanced hunting query language

Article

01/26/2023

5 minutes to read

8 contributors Feedback

In this article

Note Want to experience Microsoft 365 Defender? Learn more about how you can evaluate and pilot Microsoft 365 Defender.

Applies to:

Microsoft 365 Defender

Advanced hunting is based on the Kusto query language. You can use Kusto operators and statements to construct queries that locate information in a specialized schema.

Watch this short video to learn some handy Kusto query language basics.

To understand these concepts better, run your first query.

Try your first query

In the Microsoft 365 Defender portal, go to Hunting to run your first query. Use the following example:

```
// Finds PowerShell execution events that could involve a download union DeviceProcessEvents, DeviceNetworkEvents | where Timestamp > ago(7d) // Pivoting on PowerShell processes | where FileName in~ ("powershell.exe", "powershell_ise.exe") // Suspicious commands | where ProcessCommandLine has_any("WebClient", "DownloadFile", "DownloadData", "DownloadString", "WebRequest", "Shellcode", "http", "https") | project Timestamp, DeviceName, InitiatingProcessFileName, InitiatingProcessCommandLine, FileName, ProcessCommandLine, RemoteIP, RemoteUrl, RemotePort, RemoteIPType | top 100 by Timestamp
```

Run this query in advanced hunting

Describe the query and specify the tables to search

A short comment has been added to the beginning of the query to describe what it

is for. This comment helps if you later decide to save the query and share it with others in your organization.

```
// Finds PowerShell execution events that could involve a download
```

The query itself will typically start with a table name followed by several elements that start with a pipe (|). In this example, we start by creating a union of two tables, DeviceProcessEvents and DeviceNetworkEvents , and add piped elements as needed.

```
union DeviceProcessEvents, DeviceNetworkEvents
```

Set the time range

The first piped element is a time filter scoped to the previous seven days. Limiting the time range helps ensure that queries perform well, return manageable results, and don't time out.

```
| where Timestamp > ago(7d)
```

Check specific processes

The time range is immediately followed by a search for process file names representing the PowerShell application.

```
// Pivoting on PowerShell processes | where FileName in~ ("powershell.exe", "powershell_ise.exe")
```

Search for specific command strings

Afterwards, the query looks for strings in command lines that are typically used to download files using PowerShell.

```
// Suspicious commands | where ProcessCommandLine has_any("WebClient", "DownloadFile", "DownloadData", "DownloadString", "WebRequest", "Shellcode", "http", "https")
```

Customize result columns and length

Now that your query clearly identifies the data you want to locate, you can define what the results look like. project returns specific columns, and top limits the number of results. These operators help ensure the results are well-formatted and reasonably large and easy to process.

| project Timestamp, DeviceName, InitiatingProcessFileName, InitiatingProcessCommandLine, FileName, ProcessCommandLine, RemoteIP, RemoteUrl, RemotePort, RemoteIPType | top 100 by Timestamp

Select Run query to see the results.

Tip You can view query results as charts and quickly adjust filters. For guidance, read about working with query results

Learn common query operators

You've just run your first query and have a general idea of its components. It's time to backtrack slightly and learn some basics. The Kusto query language used by advanced hunting supports a range of operators, including the following common ones.

Operator Description and usage where Filter a table to the subset of rows that satisfy a predicate. summarize Produce a table that aggregates the content of the input table. join Merge the rows of two tables to form a new table by matching values of the specified column(s) from each table. Watch Joining tables in KQL to learn how. count Return the number of records in the input record set. top Return the first N records sorted by the specified columns. limit Return up to the specified number of rows. project Select the columns to include, rename or drop, and insert new computed columns. extend Create calculated columns and append them to the result set. makeset Return a dynamic (JSON) array of the set of distinct values that Expr takes in the group. find Find rows that match a predicate across a set of tables.

To see a live example of these operators, run them from the Get started section in advanced hunting.

Understand data types

Advanced hunting supports Kusto data types, including the following common types:

Data type Description and query implications datetime Data and time information typically representing event timestamps. See supported datetime formats string Character string in UTF-8 enclosed in single quotes (') or double quotes ("). Read more about strings bool This data type supports true or false states. See supported literals and operators int 32-bit integer long 64-bit integer

To learn more about these data types, read about Kusto scalar data types.

Get help as you write queries

Take advantage of the following functionality to write queries faster:

Autosuggest â€”as you write queries, advanced hunting provides suggestions from IntelliSense.

â€”as you write queries, advanced hunting provides suggestions from IntelliSense. Schema tree â€”a schema representation that includes the list of tables and their columns is provided next to your working area. For more information, hover over an item. Double-click an item to insert it to the query editor.

â€”a schema representation that includes the list of tables and their columns is provided next to your working area. For more information, hover over an item. Double-click an item to insert it to the query editor. Schema referenceâ€”in-portal reference with table and column descriptions as well as supported event types (ActionType values) and sample queries

Work with multiple queries in the editor

You can use the query editor to experiment with multiple queries. To use multiple queries:

Separate each query with an empty line.

Place the cursor on any part of a query to select that query before running it. This will run only the selected query. To run another query, move the cursor accordingly and select Run query.

For a more efficient workspace, you can also use multiple tabs in the same hunting page. Select New query to open a tab for your new query.

You can then run different queries without ever opening a new browser tab.

Note Using multiple browser tabs with advanced hunting might cause you to lose your unsaved queries. To prevent this from happening, use the tab feature within advanced hunting instead of separate browser tabs.

Use sample queries

The Get started section provides a few simple queries using commonly used operators. Try running these queries and making small modifications to them.

Note Apart from the basic query samples, you can also access shared queries for specific threat hunting scenarios. Explore the shared queries on the left side of the page or the GitHub query repository.

Access query language documentation

For more information on Kusto query language and supported operators, see Kusto query language documentation.

Note Some tables in this article might not be available in Microsoft Defender for Endpoint. Turn on Microsoft 365 Defender to hunt for threats using more data sources. You can move your advanced hunting workflows from Microsoft Defender for Endpoint to Microsoft 365 Defender by following the steps in Migrate advanced hunting queries from Microsoft Defender for Endpoint.

Related topics

Reference

[Creativity in Research: Cultivate Clarity, Be Innovative, and Make Progress in your Research Journey]

[Bad Pharma: How Drug Companies Mislead Doctors and Harm Patients]